ESD-TR-75-360

MTR-3095

USER REQUIREMENTS LANGUAGE AND ANALYZER (URL/URA) AN INSTALLATION GUIDE

JANUARY 1976

Prepared for

DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
Hanscom Air Force Base, Bedford, Massachusetts

Project No. 572H
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract No. F19628-75-C-0001

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication

HENRY J. EIDEN, MAJ, USAF
Program Manager

MICHAEL S. FREEMAN, CAPT, USAF
Project Engineer

FOR THE COMMANDER

FRANK J. EMMA, COL, USAF
Director, ISTAO
Deputy for Command and Management Systems

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Date Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-75-360	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* USER REQUIREMENTS LANGUAGE AND ANALYZER (URL/URA) – AN INSTALLATION GUIDE		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER MTR-3095
7. AUTHOR(s) L. C. Cheng		8. CONTRACT OR GRANT NUMBER(s) F19628-75-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS The MITRE Corporation Box 208 Bedford, MA 01730		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Project No. 572H
11. CONTROLLING OFFICE NAME AND ADDRESS Deputy for Command and Management Systems Electronic Systems Division, AFSC Hanscom Air Force Base, Bedford, MA 01731		12. REPORT DATE JANUARY 1976 13. NUMBER OF PAGES 56
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15e. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

CARA PROJECT
COMPUTER AIDED REQUIREMENTS ANALYSIS
SPECIFICATION LANGUAGE

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This document contains instructions for installing URL/URA (User Requirements Language/User Requirements Analyzer) on the IBM 370/158 operating under OS/MVT and Time Sharing Option (TSO). Guidelines are also given for installation in other environments. This document is written for systems personnel who will

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Date Entered)*

be responsible for the installation and they are therefore presumed to be familiar with the terminology used.

ACKNOWLEDGMENT

This report has been prepared by The MITRE Corporation under Project 572H. The contract is sponsored by the Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

TABLE OF CONTENTS

TABLE OF CONTENTS (Concluded)

LIST OF ILLUSTRATIONS

SECTION I

A BRIEF SYSTEMS OVERVIEW

This document contains installation and startup instructions for the system User Requirements Language and User Requirements Analyzer (URL/URA). URL/URA is an interactive computer system acquired by Air Force ESD/MCI from the University of Michigan under the task of Computer Aided Requirements Analysis (CARA).

URL/URA presently operates on the IBM 370/158 under the OS/MVT/TSO environment, and has been shown to be operable under both MVT and VS. The source code is written mainly in FORTRAN, compilable under either the G or the H compiler. A small number of routines, however, are written in Assembler (F or G).

Distribution of this system in object deck form consists of over sixty data sets, and in source form about another sixty data sets. Because of the complexity of the interrelationships among these modules, it is felt that an exposition of the overall view of the system organization is in order before detailed installation instructions are given. A brief description of the system organization will therefore be given in this section. It is intended as background information to improve understanding of the installation procedure but by no means meant to supplant any system documentation of URL/URA.

As is indicated by the name, URL/URA consists of two main parts, namely, the language and the analyzer. URL is a language for formally stating user requirements. A target system can be described by statements of the URL language which can be read and constructed into a data base via a set of interactive commands. This data base can in turn be checked, analyzed and summarized by another set of commands. The analyzer (URA) is the set of modules or computer programs that make up these commands. Hence, installing the URL/URA system is in reality installing the URA programs as well as supportive software to facilitate their execution.

URA COMMANDS AND CORRESPONDING PROGRAMS

Listed below are the names of all the program modules that make up the URA commands and the specific commands to which they correspond.

Program Module	URA Command
CM	CM (CONSISTS-MATRIX)
CNC	CNC (CONSISTS-COMPARISON)

Program Module	URA Command
CONT	CONT (CONTENTS)
CT	CT (CHANGE-TYPE)
DBS	DBS (DATA-BASE-STATISTICS)
DCOM	DCOM (DELETE-COMMENT-ENTRY)
DEL	DEL (DELETE)
DICT	DICT (DICTIONARY)
DP	DP (DATA-PROCESS)
DPSL	DPSL (DELETE-PSL)
EI	EI (ENTITY-IDENTIFIER)
FPS	FPS (FORMATTED-PROBLEM-STATEMENT)
FREQ	FREQ (FREQUENCY)
IDX	INDEX option in PIC, STR & FPS
KPER	KWIC (KWIC-INDEX)
KPRT	
NGA	NG (NAME-GEN)
NGP	
NLA	NL (NAME-LIST)
NLP	
PAV	PAV (PRINT-ATTRIBUTE-VALUES)
PCOM	PCOM (PUNCH-COMMENT-ENTRY)
PIC	PIC (PICTURE)
PRIO	PRIO (PROCESS-INPUT-OUTPUT)
RCOM	RCOM (REPLACE-COMMENT-ENTRY)
REN	REN (RENAME)
STR	STR (STRUCTURE)
SUM	SUM (SUMMARY)
SYNU	IP (INPUT-PSL)
XREF	cross reference option for IP

These programs are all written in FORTRAN. They are, however, not complete by themselves. They are supplemented by two other types of data sets - block data and service routine libraries.

BLOCK DATA

There are altogether four block data programs, namely,

PSL
PSLBLK
LIBBLK
and DBBLK.

These are all FORTRAN routines containing data statements which give initial values to variables in COMMON blocks. Not all four of

the block data programs are required by the URA command program modules listed above. SYNU and DPSL require PSL and PSLBLK. All the others require LIBBLK and DBBLK.

LIBRARIES

Libraries contain common subroutines used by more than one command program; they also contain service subroutines that are used over and over again. These libraries are compiled separately and link-edited together with the compiled program modules to form complete load modules.

There are three levels of libraries in support of the URA software:

1. SLIB - routines that are either operating system or machine dependent. These are mainly written in assembly language.

2. DBLIB - data base routines. These are routines that make up the data base management section of the software. Most of the routines are written in FORTRAN. The random access I/O routines, however are written in Assembler. Some routines in DBLIB reference routines in SLIB.

3. FLIB - These are general utility FORTRAN subroutines used throughout the analyzer. Examples are report routines and matrix-handling routines. Since these routines are written in FORTRAN, they should be independent of the machine or operating system environment, except for the variations of the FORTRAN language as implemented in the different machines. The routines in FLIB reference routines in DBLIB and SLIB.

TOP-LEVEL OPERATING ENVIRONMENT

So far we have identified the program modules which compose the analyzer commands and their supportive software. The question remains: how are these commands initiated? This is achieved by a top-level command language handler which reads in parameters from users, accepts URA commands and initiates the correct program module(s) to handle each specific command. This top-level command language handler consists of two parts - CLI, the main portion, and CLIEX, which contains most of the code likely to be changed.

The entire systems overview is summarized in Figure 1.

This particular version of URA being distributed operates under the IBM 370 Time Sharing Option (TSO). The mechanism whereby the program modules are activated is TSO command procedures. First, CLI is initiated via a command procedure CLI.CLIST. CLI accepts URA commands and their respective parameters, constructs another command procedure GO.CLIST which is executed via a statement in CLI.CLIST (see listing of CLI.CLIST in Figure 2). CLI.CLIST is automatically reactivated by the last statement in GO.CLIST, and another URA command can be accepted by CLI. This mode of operation is very dependent on some subtle characteristics of TSO command procedures. Its pitfalls and alternatives will be addressed later.

AUXILIARY URA DATA SETS

Besides the URA command software discussed above, there is another category of data sets which is required before the URA system can operate. Examples of these data sets are data base tables required for the initialization of the data base and system default parameters which can be changed at each installation.

Listed below are the names of such data sets and their functions.

- PSADDL.DATA - data base description.
  Required for the initialization of a data base.
- SECRET.DATA - system default parameters.
  Can be changed at each installation.
- HELPSHRT.DATA - text for the HELP command (short version).
- HELPLONG.DATA - text for the HELP command (long version).
- PSAERROR.DATA - text for URA error message.
- DDLA - data base description analysis program.
  It generates database tables from database description.
  These tables are required for database initialization.
- DBIN - database initialization program.

SYSTEM SUPPORT DATA SETS

Some data sets are not part of URA itself but are included in the distribution to facilitate the installation of the URA system. One example is the INCL program which expands the Macro-like calls in the

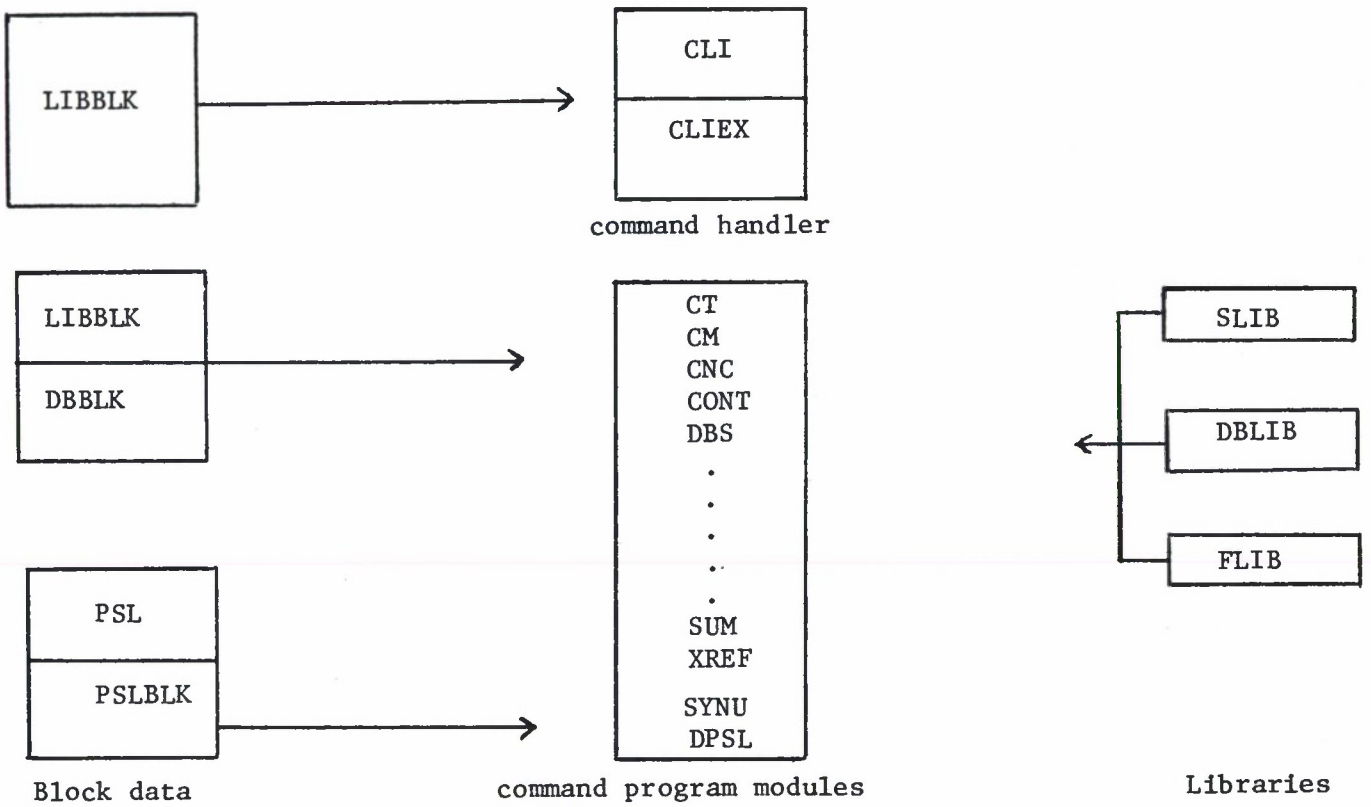

Figure 1. URA Command Modules

```
 CLI.CLIST
00010 PROC 0 DBN(URADB.DATABASE) OUT(*) MOD
00020 FREE FI(FT10F001,FT11F001,FT13F001,FT15F001,FT16F001,FT17F001,FT18F001)
00030 FREE FI(FT19F001,FT20F001,FT06F001,FT12F001) ATTR(VBS,FTA)
00040 ATTR VBS RECFM(V B S) BLKSIZE(800)
00050 ATTR FBA RECFM(F B A)
00060 ALLOC F(FT06F001) DA(*)
00070 ALLOC F(FT10F001) DA(*)
00080 ALLOC F(FT11F001) DA(*) USING(FBA)
00090 ALLOC F(FT13F001) DA(URAFPAR.UNFORM) USING(VBS)
00100 ALLOC F(FT15F001) DA('TS0448.URAERROR.DATA') SHR
00110 ALLOC F(FT16F001) DA('TS0448.HELPLONG.DATA') SHR
00120 ALLOC F(FT17F001) DA('TS0448.HELPSHRT.DATA') SHR
00130 ALLOC F(FT18F001) DA('TS0448.SECRET.DATA') SHR
00140 ALLOC F(FT19F001) DA(GO.CLIST)
00150 ALLOC F(FT20F001) DA(SORT.DATA)
00155 ALLOC F(FT12F001) BLOCK(800) SPACE(10)
00160 CALL 'TS0448.CALLIB.LOAD(CLI)'
00170 FREE DA('TS0448.CALLIB.LOAD')
00180 FREE FI(FT10F001,FT11F001,FT13F001,FT15F001,FT16F001,FT17F001,FT18F001)
00190 FREE FI(FT19F001,FT20F001,FT12F001) ATTR(VBS,FBA)
00200 EXEC GO.CLIST 'DBN(&DBN.) OUT(&OUT.) &MOD.'
00210 END
READY
```

Figure 2. Listing of CLI.CLIST

FORTRAN source code (see Section V). Others will not be enumerated here but will be discussed as they are introduced.

SECTION II

DISTRIBUTION TAPES

Data sets on the URA distribution tape are all IEHMOVE unloaded data sets, i.e., they are all copied from disk storage to tape by the IBM utility program IEHMOVE. Standard labels are used and recording done at 1600 b.p.i. Record format is fixed blocked, record length is 80 and block size is 800 characters.

Contents of the tape not only include the various types of data sets described in the previous section, i.e. the data sets that make up URA proper, but also a group of miscellaneous data sets which are used to help install the URA system. These data sets will no longer be required as soon as the system is in operation.

Naming conventions used are standard IBM 370 TSO conventions. The qualifier .FORT denotes FORTRAN source data sets (e.g. CLI.FORT, CT.FORT), .ASM denotes assembler source code, .OBJ denotes object modules, .LOAD denotes load modules, .CLIST denotes TSO command procedures and .DATA denotes 80 character card images. All data set names on the tape are prefixed by a TSO account number which should be changed to what is appropriate at each installation.

The URA tape consists of object and load modules. For facilities with IBM 360 or 370 computers operating under OS/MVT or VS, the URA software should be installed from load modules. For versions of these operating systems with SVC's or access methods not compatible with those of OS/MVT, installation should be done using object modules after rewriting the random access routines in the URA software. The instructions given in this document are intended for the TSO environment. For installations which do not operate under TSO, appropriate modifications should be made to the instructions.

The following Table is a listing of the data sets on the tape in the order they reside on the tape.

For the benefit of those who require access to the source. code, a URA source tape is also available. The contents of the tape will be listed in Section V. An attempt will also be made in Section V to highlight certain considerations in installing the system from source code. However, no detailed instructions are given, for it is felt that the installation procedure is too complex and environment-dependent to implement without adequate system documentation or assistance both from people familiar with the systems aspects of URA, and people familiar with the system environment of the proposed installation.

Table I

Data Sets on Object Distribution Tape

Data Set Number	Data Set Name
1	TSO448.CALLIB.LOAD
2	TSO448.PSADDL.DATA
3	TSO448.SECRET.DATA
4	TSO448.HELPSHRT.DATA
5	TSO448.HELPLONG.DATA
6	TSO448.PSAERROR.DATA
7	TSO448.URA.CLIST
8	TSO448.CLI.CLIST
9	TSO448.DDLA.CLIST
10	TSO448.DBIN.CLIST
11	TSO448.PSALINK.CLIST
12	TSO448.PSALINK1.CLIST
13	TSO448.NCLI.CLIST
14	TSO448.LIBMTS.FORT
15	TSO448.DBRAND.ASM
16	TSO448.DBLIB.LOAD
17	TSO448.FLIB.LOAD
18	TSO448.SLIB.LOAD
19	TSO448.CLI.OBJ
20	TSO448.CLIEX.OBJ
21	TSO448.CM.OBJ
22	TSO448.CNC.OBJ
23	TSO448.CT.OBJ
24	TSO448.CONT.OBJ
25	TSO448.DBS.OBJ
26	TSO448.DCOM.OBJ
27	TSO448.DEL.OBJ
28	TSO448.DICT.OBJ
29	TSO448.DP.OBJ
30	TSO448.DPSL.OBJ
31	TSO448.EI.OBJ
32	TSO448.FPS.OBJ
33	TSO448.FREQ.OBJ
34	TSO448.IDX.OBJ
35	TSO448.KPER.OBJ
36	TSO448.KPRT.OBJ
37	TSO448.NGA.OBJ
38	TSO448.NGP.OBJ
39	TSO448.NLA.OBJ
40	TSO448.NLP.OBJ

Table I

Data Sets on Object Distribution Tape (Concluded)

Data Set Number	Data Set Name
41	TSO448.PAV.OBJ
42	TSO448.PCOM.OBJ
43	TSO448.PIC.OBJ
44	TSO448.PRIO.OBJ
45	TSO448.RCOM.OBJ
46	TSO448.REN.OBJ
47	TSO448.SUM.OBJ
48	TSO448.SYNU.OBJ
49	TSO448.XREF.OBJ
50	TSO448.DBIN.OBJ
51	TSO448.DDLA.OBJ
52	TSO448.PSL.OBJ
53	TSO448.PSLBLK.OBJ
54	TSO448.LIBBLK.OBJ
55	TSO448.DBBLK.OBJ
56	TSO448.EXA.DATA
57	TSO448.EXB.DATA

SECTION III

INSTALLATION FROM LOAD MODULES

All the load modules are grouped in one partitioned data set called TSO448.CALLIB.LOAD.

COPY DATA SETS FROM TAPE

All the following data sets should be copied from the object tape to disk (or other direct access storage) with the IBM 370 utility program IEHMOVE. The data sets should be renamed replacing the account number by the TSO account number at your installation.

Data Set Number on Object Tape	Data Set Name
1	TSO448.CALLIB.LOAD
2	TSO448.PSADDL.DATA
3	TSO448.SECRET.DATA
4	TSO448.HELPSHRT.DATA
5	TSO448.HELPLONG.DATA
6	TSO448.PSAERROR.DATA
7	TSO448.URA.CLIST
8	TSO448.CLI.CLIST
9	TSO448.DDLA.CLIST
10	TSO448.DBIN.CLIST
56	TSO448.EXA.DATA
57	TSO448.EXB.DATA

MODIFY TSO COMMAND PROCEDURES

Inspect all the .CLIST data sets, changing all occurrences of TSO448 to the new account number as well as checking to make sure each command procedure will work at your installation.

SET DEFAULT PARAMETERS

SECRET.DATA contains default parameters which can be changed for your particular installation. Again, all occurrences of TSO448 should be changed to your own account number.

INITIALIZE A DATA BASE

A data base is created or initialized in two steps. First of all, a data base description PSADDL.DATA is analyzed by a program

DDLA and transformed into a data base table PSADBTAB.UNFORM. The data base table is then formatted by a program DBIN to form a data base which can be interfaced with the rest of the system. (See Figure 3.) Both of these programs are included in load module form in CALLIB.LOAD.

The TSO commands to achieve the above are contained in two command procedures -- DDLA.CLIST and DBIN.CLIST. These should be executed by the following commands.

```
EXEC  DDLA  'PSADDL.DATA  PSADBTAB.UNFORM'
EXEC  DBIN  'PSADB.DATABSE  PSADBTAB.UNFORM'
```

A listing of the two CLIST's can be found in Figure 4.

The first line of the data base description contains the number of pages allowable for the data base to be initialized (1 page is equivalent to 4K bytes). By varying this number, the size of the data base can be controlled. The number of pages in the version of PSADDL.DATA being distributed is 50. However, if this number is changed, the space allocated for the data base in DBIN.CLIST should also be changed accordingly.

Instead of using PSADDL.DATA, one can write one's own data base description. The data base structure is documented in detail in "The Structure and Contents of a PSA Data Base"(1) and "A Data Base Management System for PSA Based on DBTG 71"(2).

TO START UP THE SYSTEM

For system startup, first enter TSO command:

```
EXEC  URA
```

This generates all temporary data sets required for execution of URA commands. Then enter:

```
EXEC  CLI
```

which is a restart procedure that brings the execution into URA command mode - that is, URA commands can be entered. If at any time an error occurs such that execution is taken out of this mode back into TSO command mode, CLI.CLIST can be executed again by:

```
EXEC  CLI
```

to restart the URA system.

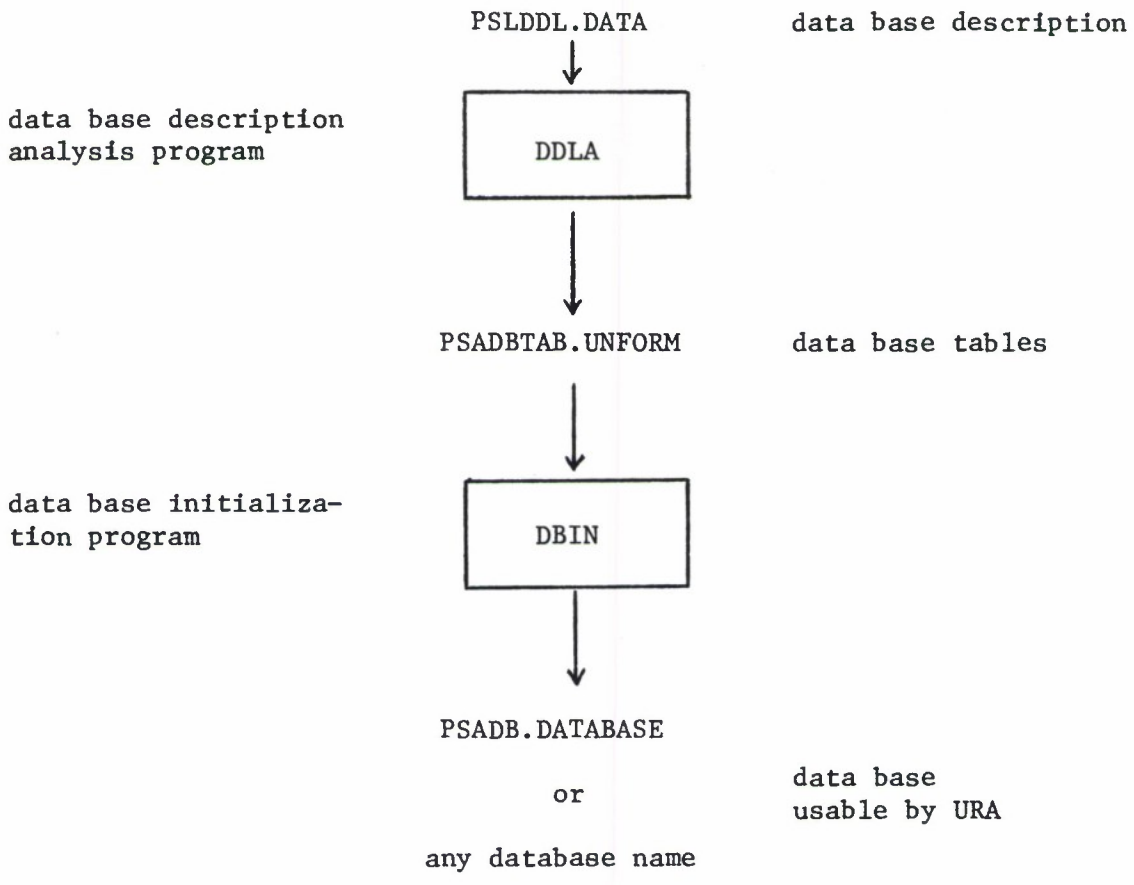

Figure 3. Initialization of a URA Data Base

```
 DDLA.CLIST
00010 PROC 2 DDL DBT PRINT(*)
00020 FREE FI(FT05F001,FT06F001,FT07F001,FT08F001)
00030 ALLOC F(FT05F001) DA(&DDL.)
00040 ALLOC F(FT06F001) DA(&PRINT.)
00050 ALLOC F(FT07F001) DA(*)
00060 FREE ATTR(UNFORM)
00070 ATTR UNFORM RECFM(V B S) BLKSIZE(800)
00080 FILESTAT &DBT
00090 SYSRC (EQ 0) DELETE &DBT
00100 ALLOC F(FT08F001) DA(&DBT.) NEW SPACE(10,10) BLOCK(800) USING(UNFORM)
00110 CALL 'TS0448.CALLIB.LOAD(DDLA)'
00120 FREE ATTR(UNFORM)
00130 FREE FI(FT05F001,FT06F001,FT07F001,FT08F001)
READY
```



```
 DBIN.CLIST
00010 PROC 2 DBF DBT PRINT(*)
00020 FREE FILE(DB0002,FT03F001,FT06F001)
00030 ALLOC F(FT03F001) DA(&DBT.)
00040 FILESTAT &DBF
00050 SYSRC (EQ 0) DELETE &DBF
00060 ALLOC F(DB0002) DA(&DBF.) NEW SPACE(30,10) BLOCK(4096)
00070 ALLOC FI(FT06F001) DA(&PRINT.)
00080 CALL 'TS0448.CALLIB.LOAD(DBIN)'
00090 FREE FILE(FT03F001,DB0002)
READY
```

Figure 4. CLIST's for Initialization of Data Base

As explained in Section I, the ability to enter another URA command as soon as the previous one has finished execution depends on recursive executions of the two command procedures CLI.CLIST and GO.CLIST. However, each such recursive initiation of a CLIST from another CLIST uses up a certain amount of core storage. If enough URA commands have been executed without interruption, one could run out of core. In this case, the URA system can also be restarted just by executing CLI.CLIST again.

The newly installed URA system can be tested with the two examples included (EXA.DATA and EXB.DATA). These contain URL statements which can be entered into the data base.

SECTION IV

LOADING FROM OBJECT MODULES

If the URA system is being installed on the IBM 370 or a comparable machine but the operating system is not altogether compatible with OS/MVT, the procedures outlined in this section should be followed.

COPY DATA SETS FROM TAPE

All data sets except CALLIB.LOAD should be copied from the object tape to disk with the IBM 370 utility program IEHMOVE. The data sets should be renamed replacing the account number by the TSO account number at your installation.

MODIFY TSO COMMAND PROCEDURES

Inspect all the .CLIST data sets, changing all occurrences of TS0448 to the new account number as well as checking to make sure each CLIST will work at your installation.

INSTALLATION-DEPENDENT CODE

The code that most likely needs to be changed or rewritten is the code that is special to each installation. The installation dependent code in URA resides within two source libraries DBRAND.ASM and LIBMTs.FORT.

The routines in DBRAND.ASM, viz:

RANDCL
RANDOP
RANDOW
RANDRD
RANDRW
RANDWT

are random access routines. A listing of their contents can be found in Appendix I. If not compatible with the operating system at your particular institution, they should be rewritten. Each routine should then be compiled and link-edited into a separate load module which should replace the routine of the same name in the library DBLIB.LOAD.

The source library LIBMTS.FORT has the following members, listings of which can also be found in Appendix I:

CLOSEDB
CPUSEC
GETDAT
GETERR
GETTIM

No change is required for CLOSEDB and GETERR. The other routines all make use of Assembly subroutines which obtain the date, time of day and CPU time. These subroutines are called INTIME, JTIME, DATE and CLOCK. A listing of their source code and an explanation of their usage can be found in Appendix II. The source code should be scanned to see if they are compatible with your operating system. If so, no change is required. Otherwise, the code should be modified so that it will execute correctly in your facility. Or, there may be similar timing routines in your installation which will serve the purpose. In that case, the FORTRAN instructions in CPUSEC, GETDAT and GETTIM which call these timing routines should be changed. In either case, the three routines, CPUSEC, GETDAT and GETTIM have to be recompiled and a load module for each including new versions of the timing subroutines. Each of these load modules CPUSEC, GETDAT and GETTIM should replace members of the same name in the load library SLIB.LOAD.

RE-LINK-EDIT

Since any of the changes outlined above involves a change in the libraries DBLIB or SLIB, the URA command handler and command program modules have to be re-link-edited to include the new versions of the library modules (refer to Figure 1).

The command handler CLI.OBJ should be link-edited using the command procedure NCLI.CLIST, SYNU.OBJ and DPSL.OBJ should be link-edited using PSALINK1.CLIST and all other URA command program modules such as CT.OBJ, CM.OBJ and XREF.OBJ with PSALINK.CLIST. Each of the resulting load modules will automatically replace members of the same name in CLLIB.LOAD. The two programs required in the initialization of a data base (DBIN.OBJ and DDLA.OBJ) should also be link-edited in a similar way with PSALINK.CLIST.

At this point, the procedures set forth in the previous section under paragraphs Set Default Parameters, Initialize a Data Base and To Startup the System should be followed to complete the installation.

SECTION V

SOURCE MODULES

For the benefit of those who would like to make changes to the URL/URA system, or those who would like to install it on a machine other than the IBM 360/370 series, the URA system is distributed in source form as well. Table II contains a list of the data sets on the source distribution tape. A "C" denotes those data sets which are of interest only to those making changes to an already installed URA system operating under TSO on the IBM 360/370.

Note that if the system were to be installed on a machine whose tape recording format is not compatible with that of the IBM 360/370 on which the distribution tape is made, the source code will have to be distributed via card decks or whatever other means which makes the transfer possible. As has already been mentioned, installation of this system on another machine without adequate systems documentation or the help of those familiar with the system is really not advisable. Besides, no detailed instructions can be given here without knowing the environment under which the system will operate. However, an attempt will be made in this section to point out the peculiarities that have to be taken into account and to outline the steps that should be taken. Making changes to the source code on the same machine (IBM 360/370) is much simpler and the procedure will also be outlined.

Most of the source code is written in FORTRAN, the rest in 370 Assembly Language.

FORTRAN CONVENTIONS

An unusual convention similar to a macro call is used in the FORTRAN source code. The rationale behind this is that there are many common sections of code which are used throughout the URA programs. This common code consists mainly of FORTRAN COMMON blocks which define variables shared among subroutines, and FORTRAN DATA statements which set the values of these variables. Furthermore, these variables often represent parameters such as the size of the symbol table, number of lines in a page etc., which users of URA may wish to change from time to time. These common blocks of code are therefore not included in the FORTRAN source, but their required presence is indicated by the name of the block. For example, if

```
*CALL,RECTAB
```

appears in between two FORTRAN statements, a common block called RECTAB should be inserted in its place. All the common blocks of code to be

Table II

Data Sets for Source Distribution

Data Set Number	Data Set Name
1	TSO448.PSADDL.DATA
2	TSO448.SECRET.DATA
3	TSO448.HELPSHRT.DATA
4	TSO448.HELPLONG.DATA
5	TSO448.PSAERROR.DATA
6	TSO448.DDLA.DATA
7	TSO448.DBIN.DATA
8	TSO448.INCL.DATA
9	TSO448.DBCOM.DATA
10	TSO448.PSACOM.DATA
11	TSO448.EXA.DATA
12	TSO448.EXB.DATA
13	TSO448.SLIB.ASM
14	TSO448.DBASM.ASM
15	TSO448.DBRAND.ASM
16	TSO448.LIBMTS.FORT
17	TSO448.DBUSER.DATA
18	TSO448.DBLOW.DATA
19	TSO448.DBTAB.DATA
20	TSO448.FLIB.DATA
21	TSO448.PSL.DATA
22	TSO448.PSLBLK.DATA
23	TSO448.DBBLK.DATA
24	TSO448.LIBBLK.DATA
25	TSO448.CLI.DATA
26	TSO448.CLIEX.DATA
27	TSO448.CM.DATA
28	TSO448.CNC.DATA
29	TSO448.CONT.DATA
30	TSO448.CT.DATA
31	TSO448.DBS.DATA
32	TSO448.DCOM.DATA
33	TSO448.DEL.DATA
34	TSO448.DICT.DATA
35	TSO448.DP.DATA

Table II

Data Sets for Source Distribution (Concluded)

Data Set Number	Data Set Name	
36	TSO448.DPSL.DATA	
37	TSO448.EI.DATA	
38	TSO448.FPS.DATA	
39	TSO448.FREQ.DATA	
40	TSO448.IDX.DATA	
41	TSO448.KPER.DATA	
42	TSO448.KPRT.DATA	
43	TSO448.NGA.DATA	
44	TSO448.NGP.DATA	
45	TSO448.NLA.DATA	
46	TSO448.NLP.DATA	
47	TSO448.PAV.DATA	
48	TSO448.PCOM.DATA	
49	TSO448.PIC.DATA	
50	TSO448.PRIO.DATA	
51	TSO448.RCOM.DATA	
52	TSO448.REN.DATA	
53	TSO448.STR.DATA	
54	TSO448.SUM.DATA	
55	TSO448.SYNU.DATA	
56	TSO448.XREF.DATA	
57	TSO448.PSACOM.DATABASE	C
58	TSO448.FORTIHC.CLIST	C
59	TSO448.LIBBLK.OBJ	C
60	TSO448.DBBLK.OBJ	C
61	TSO448.PSLBLK.OBJ	C
62	TSO448.PSL.OBJ	C
63	TSO448.DBLIB.LOAD	C
64	TSO448.FLIB.LOAD	C
65	TSO448.SLIB.LOAD	C

thus inserted (a total of about 2000 lines) are organized into a data set named PSACOM.DATA. If any of these blocks were to be changed, the modification needs to take place only once in PSACOM.DATA, instead of at all occurrences of the common block throughout the code which may be quite tedious. Another advantage is that the stored source code is shorter in length. This may be important since storage requirements for URA are not trivial, as will be made evident in the next section. These advantages are achieved at the sacrifice of processing time, of course.

A FORTRAN program is written to insert the blocks from PSACOM.DATA into the source code by replacing each *CALL statement with the correct block. This program is called INCL and its source is contained in INCL.DATA. It also exists in executable load module form as a member of CALLIB.LOAD. The way the INCL program works is that it makes use of the URA data base handling capability. PSACOM.DATA has to be built into a URA data base. INCL will read in the data base as input and expand the *CALL statements into their respective COMMON blocks. A version of this data base built from PSACOM.DATA is included in this distribution of the source code for the purposes of those who already have a URA system working but wish to make modifications to the system. This will save them the trouble of having to generate the data base themselves (for instructions on building a data base please refer to Section IV). This is true provided that no change is made to the COMMON blocks in PSACOM.DATA. As can be expected, to install URA on another machine is much more complicated. We will go into that further in a subsequent paragraph.

TO MAKE CHANGES TO URA COMMAND MODULES

If some version of URA has already been installed from load or object modules, and changes are to be made in URA command program modules, the procedure is not too complicated. First, make the change to the source of the command module, say SUM.DATA. Then include the common blocks (or expand the *CALL statements) and compile the expanded FORTRAN code by executing FORTIHC.CLIST (see Figure 5).

TO MAKE CHANGES TO THE LIBRARIES

To make a change to the libraries is slightly more involved. The source for SLIB is in SLIB.DATA, that for FLIB in FLIB.DATA. However, there are several source libraries which make up DBLIB. They are:

DBLOW.DATA	DBASM.ASM
DBTAB.DATA	DBRAND.ASM
DBUSER.DATA	
LIBMTS.FORT	

```
 'TS0448.FORTIHC.CLIST'
00010 PROC 1 NAME PRINT(SYSOUT) DB('TS0448.PSACOM.DATABASE')
00015 TIME
00020 FREE FI(DB0002,FT03F001,FT05F001,FT06F001,FT07F001,FT08F001)
00030 ALLOC F(DB0002) DA(&DB.)
00040 ALLOC F(FT03F001) DA('TS0448.MODDBT.UNIFORM')
00050 ALLOC F(FT05F001) DA(&NAME..DATA)
00060 ALLOC F(FT06F001) DA(*)
00070 ALLOC F(FT07F001) DA(TEMP.FORT)
00080 ALLOC F(FT08F001) SYSOUT
00090 CALL 'TS0448.CALLIB.LOAD(INCL)'
00100 FREE FI(DB0002,FT03F001,FT05F001,FT07F001)
00105 TIME
00110 FREE FI(FT08F001) SYSOUT(T)
00120 FREE FI(SYSPRINT)
00130 ALLOC F(SYSPRINT) &PRINT
00140 FREE FILE(SYSLIN,SYSIN,SYSUT2)
00150 FILESTAT &NAME..OBJ
00160 SYSRC (EQ 0) DELETE &NAME..OBJ
00170 ALLOC FILE(SYSIN) DA(TEMP.FORT) SHR
00180 ALLOC FILE(SYSLIN) DA(&NAME..OBJ) SPACE(200,100) BLOCK(400)
00185 ALLOC FILE(SYSUT2) BLOCK(400) SPACE(100,25)
00190 CALL 'SYS1.LINKLIB(IEKAA00)' 'SOURCE,NOLIST,MAP,XREF,NOID'
00200 FREE FILE(SYSLIN,SYSIN,SYSUT2)
00210 FREE FI(SYSPRINT) SYSOUT(T)
00215 TIME
00220 END
READY
```

Figure 5. Listing of FORTIHC.CLIST

If a change is made to an Assembly Language library, one only needs to reassemble it after the change. If the change is made to a FORTRAN library, one needs to expand the *CALL statements using INCL before recompiling it. After the reassembly or recompilation, one has a library in the form of a large object deck with each CSECT or subroutine stacked one after the other. This is still not in usable form. Another support program called LKIN, the source of which is in LKIN.FORT, was written to split up the object module by subroutines. A load module form of LKIN also exists as a member in CALLIB.LOAD. After executing LKIN, the new object library produced should be link-edited into its load module library.

TO INSTALL URA ON ANOTHER MACHINE

*CALL's in the problem arise because the source FORTRAN code has to be expanded and yet to achieve this with the INCL program requires that part of the URA system, viz. the data base package, be operational. This is sort of a "bootstrapping" procedure. There are two alternatives. One is to write one's own program to expand the *CALL statements. However, considering the fact that PSACOM.DATA consists of over a hundred COMMON blocks amounting to about 2000 records, and the frequency of occurrence of the *CALL statements, this will be no easy matter. The other alternative is to bootstrap the system by first making the data base package operational. This can be done by expanding the FORTRAN code which makes up DBLIB either manually or by developing your own program to do it. Fortunately, only a small portion of PSACOM.DATA is required in the expansion of the data base libraries, and they have been extracted into a data set DBCOM.DATA. Once DBLIB is operational, INCL and DBIN can be used (their source code has to be expanded manually, too) to build PSACOM.DATA into a data base PSACOM.DATABASE and expand the source code for the rest of the system. After that, the system can be compiled, and transformed into executable (load) modules. Needless to say, none of the TSO capabilities, such as CLIST's, apply. The top level initiation of the URA command by means of TSO command procedures will no longer function in a non-TSO environment. (Refer to Top Level Operating Environment in Section I, and To Startup the System in Section III.) Another method of management will have to be worked out. Since the CLI module writes out the appropriate TSO command procedure for the URA command entered, the code in CLI will have to be modified to achieve a change in the top-level management.

SECTION VI

STORAGE REQUIREMENTS

MAIN STORAGE

This particular version of URL/URA presently operates within a 256K partition under TSO. Actually, most of the URA commands will operate within 128K, except IP and DPSL, the two commands that deal with entering and deleting information in the data base. The module that handles the IP command is called SYNU (see Section I).

The maximum number of pages (4K bytes) of the data base that can be brought into core at any time has been cut down to 4K so that SYNU can fit into a 256K partition. In building large data bases, this maximum number of pages in core can be made larger if a larger partition is available, or, if the IP command can be made to run under a batch environment with more core available. This will greatly enhance the execution speed and can be achieved by changing variable MPICOR in PSLBLK to the desired number of pages (not to exceed 30 pages) and the dimension of PAGE to a value equal to the product of MPICOR and 1024. (See Figure 6). Figure 7 contains an example of how IP can be run as a batch job on the IBM 370/158.

The operating environment to which all the above comments pertain is the OS/MVT operating environment. For operations under VS or other systems with virtual memory the partition size will no longer become a problem.

SECONDARY STORAGE

IBM 3330 disk storage is used at the installation where the distribution tapes were made. Track length is 13030 bytes. The whole URL/URA system including source, load and some object modules requires over 2000 tracks to store. Just that part of it that is required for the everyday execution of URA (only load modules, auxiliary data sets, CLISTs etc.) requires about 800 tracks. As one can see these numbers are not trivial and may incur considerable day to day operation cost.

Appendix III contains more details on the size of each data set.

OTHER

The number of files allocated by URA to service some commands may be as high as 14. Provision therefore must be made in the number of dynamic allocation of files in the TSO logon procedure.

```
02490 C
02500       COMMON/BLENS /RDBLEN,IDBLEN,SDBLEN,ODBLEN,MDBLEN,
02510      & PHILEN,PRHLEN
02520       INTEGER RDBLEN,IDBLEN,SDBLEN,ODBLEN,MDBLEN,
02530      & PHILEN,PRHLEN
02540 C
02550       COMMON/PAGINF/NPAGES,PAGSIZ,FILE,CURPNO,MPICOR,NPICOR,NCUR,
02560      & PHICUR,PREF(32)
02570       INTEGER NPAGES,PAGSIZ,FILE,CURPNO,MPICOR,NPICOR,NCUR,
02580      & PHICUR,PREF
02590 C
02600       COMMON/PAGE  /PAGE(3072)
02610       INTEGER PAGE
02620 C
02630       COMMON/DBSWS /OPENSW,LEVEL
02640       INTEGER LEVEL
02650       LOGICAL OPENSW
02660 C
02670 CMTS     COMMON/RANDCM/CURRP,CURWP,RMOD,WMOD,BLEN
02680 CMTS     INTEGER CURRP,CURWP,RMOD,WMOD
02690 CMTS     INTEGER*2 BLEN
02700 C
02710 C.....................................................................
02720       DATA NCW/4/
02730 C
02740       DATA RDBLEN/6/
02750       DATA IDBLEN/5/
02760       DATA SDBLEN/7/
02770       DATA ODBLEN/2/
02780       DATA MDBLEN/2/
02790       DATA PHILEN/2/
02800       DATA PRHLEN/1/
02810 C
02820       DATA NPAGES/2/
02830       DATA PAGSIZ/1024/
02840       DATA MPICOR/3/
02850       DATA CURPNO/0/
02860 C
02870       DATA OPENSW/.FALSE./
02880       DATA LEVEL/0/
02890 C
02900 CMTS     DATA RMOD/0/
02910 CMTS     DATA WMOD/Z00020000/
02920 CMTS     DATA BLEN/4096/
02930 C
02940       END
02950 *EOR
END OF DATA
```

Figure 6. Portion of PSLBLK

```
//IP#NORAD  JOB     (572G,D72,DESK,10),'CHENG L',CLASS=D,TIME=20,
//              REGION=256K
//JOBLIB    DD  DSN=TS0448.CALLIB.LOAD,DISP=SHR
//IP      EXEC  PGM=SYNU
//FT15F001  DD  DSN=TS0448.PSAERROR.DATA,DISP=OLD,
//              DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//FT11F001  DD  SYSOUT=A
//FT10F001  DD  SYSOUT=A
//DB0002  DD  DSN=TS0307.PSADB.DATABASE,DISP=OLD,
//              DCB=(RECFM=F,BLKSIZE=4096)
//FT03F001  DD  DSN=TS0448.PSADBTAB.UNFORM,DISP=OLD,
//              DCB=(RECFM=VBS,BLKSIZE=800)
//FT06F001  DD  SYSOUT=A
//FT13F001  DD  DSN=TS0448.PSAFPAR.UNFORM,DISP=OLD,
//              DCB=(RECFM=VBS,BLKSIZE=800)
//FT14F001  DD DSN=TS0448.COMMENTS.DATA,DISP=OLD,
//              DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//FT01F001  DD  DSN=TS0448.RTA.UNFORM,DISP=OLD,
//              DCB=(RECFM=VBS,BLKSIZE=800)
//FT05F001  DD  DSN=TS0307.NORAD1.DATA,DISP=OLD
//FT08F001  DD  DSN=TS0448.PSA8.TEMP,DISP=OLD,
//              DCB=(RECFM=FB,LRECL=40,BLKSIZE=400)
//FT09F001  DD  DSN=TS0448.PSA9.UNFORM,DISP=OLD,
//              DCB=(RECFM=VBS,BLKSIZE=800)
//
```

Figure 7. To Execute IP Under Batch Environment

APPENDIX I

LISTINGS OF DBRAND.ASM AND LIBMTS.FORT

```
list 'ts0409.libmts.fort(closdb)'
 'TS0409.LIBMTS.FORT(CLOSDB)'
00010       SUBROUTINE CLOSDB
00020 C LAST CHANGE - 740329 - AL HERSHEY
00030 C
00040 C SUBROUTINE TO CLOSE DATA BASE AND DO ANY OTHER STUFF
00050 C
11610       COMMON/PARIN/READSW,PUNSW,OUTSW,NAM(8),STA,DBNBUF,ABRTSW
11620       INTEGER NAM,DBNBUF
11630       LOGICAL READSW,PUNSW,OUTSW,STA,ABRTSW
11640 C
11650 C FOR GIVING PARAMETERS TO ROUTINES WHICH COULD READ
11660 C
11670 C READSW - WHETHER TO READ NAMES FROM LIOE
11680 C PUNSW - WHETHER TO PUNCH OUTPUT ON FOR FUTURE REFERENCE
11690 C OUTSW - WHETHER TO PRODUCE REPORT TYPE OUTPUT
11700 C NAM - PLACE TO STORE NAME, AND NAME IF GIVEN
11710 C STA - WHETHER STAND ALONE(.TRUE.) OR WITH CLI(.FALSE.)
11720 C DBNBUF - NUMBER TO USE FOR NBUF WHEN OPENNING D.B.
11730 C ABRTSW - WHETHER OR NOT TO STOP DEAD IN PASS TWO
11740 C
14420       COMMON/BUF80/BUF80(20)
14430       INTEGER BUF80
14440 C
14450 C 80 CHARACTER BUFFER FOR GENERAL PURPOSES
14460 C
06480       COMMON/SETS/ALLNAM(2),RELA(2),RELB(2),RELC(2),ALINE(2),
06490      & SYNFOR(2),ALPHA(2),BYVAL(2)
06500       INTEGER ALLNAM,RELA,RELB,RELC,ALINE,
06510      & SYNFOR,ALPHA,BYVAL
06520 C
06530 C CHARACTER FORM OF SET NAMES
06540 C  INITIALIZED BY BLOCK DATA
06550 C
06400       COMMON/BLANKS/BLANK(30)
06410       INTEGER BLANK
06420 C
06430 C CONTAINS BLANKS FOR VARIOUS PURPOSES
06440 C  INITIALIZED BY BLOCK DATA
06450 C
00100       INTEGER IERR,NUM,ARRAY(6),I
00110 C.....................
00120       CALL SMOVE(BUF80,1,BLANK,1,80)
00130       CALL CARD(ALLNAM,NUM,IERR)
00140       CALL DBST(63,ARRAY,IERR)
00150       CALL ITOC(DBNBUF,BUF80,1,4)
00160       CALL ITOC(NUM,BUF80,5,6)
00170       DO 10 I=1,4
00180    10 CALL ITOC(ARRAY(I),BUF80,I*5+6,5)
00190       CALL ITOC(ARRAY(5),BUF80,31,7)
00200       CALL ITOC(ARRAY(6),BUF80,38,5)
00210 C     CALL PUTSTA('SELV:TPSASTAT(LAST+1) ',42,BUF80)
00220       CALL CLOS(0,ARRAY,IERR)
00230       RETURN
00240       END
READY
```

```
list 'ts0409.libmts.fort(cpusec)'
 'TS0409.LIBMTS.FORT(CPUSEC)'
00250       SUBROUTINE CPUSEC(S)
00260 C
00270 C     IF S = 0 ,   INITIALIZE
00280 C
00290 C     OTHERWISE, SET S EQUAL TO THE NUMBER OF CPU MILLISECONDS
00300 C        USED SINCE LAST CALL TO CPUSEC WITH S = 0
00310 C
00320       INTEGER S,K,I
00330 C
00340 C     VALUE RETURNED...
00350 C
00360 C     S -- ELAPSED CPU SECONDS SINCE INITIALIZATION
00370 C
00380       IF(S)20,10,20
00390    10 CALL INTIME(0)
00400       RETURN
00410 C
00420    20 I=JTIME(S,K)
00425       S = S * 26.04166 /1000.
00430       RETURN
00440       END
READY
```

```
list 'ts0409.libmts.fort(getdat)'
 'TS0409.LIBMTS.FORT(GETDAT)'
00450       SUBROUTINE GETDAT(MDY)
00460 C LAST CHANGE - 740320 - AL HERSHEY
00470 C
00480 C RETURN DATE IN CHARACTERS
00490 C
00500       INTEGER MDY(1)
00505       DATA BLANKS/'    '/
00510 C
00520 C MDY -> 12 CHARACTER DATE (MMM DD, 19YY IN MTS)
00530 C
00540 C.....................
00550       CALL DATE(MDY)
00555       MDY(3)=BLANKS
00560       RETURN
00570       END
READY
```

```
llst 'ts0409.llbmts.fort(geterr)'
 'TS0409.LIBMTS.FORT(GETERR)'
00010       SUBROUTINE GETERR(ERRNUM,ERRLEN,ERRCH)
00020 C LAST CHANGE MTS VERSION - 740322 - AL HERSHEY
00030 C
00040 C GET CHAR FORM OF ERROR FROM ERROR FILE
00050 C
00060       INTEGER ERRNUM,ERRLEN,ERRCH(1)
00070 C
00080 C ERRNUM -> ERROR NUMBER
00090 C ERRLEN <- LENGTH OF ERROR COMMENT RETURNED
00100 C ERRCH <- CHARACTER FOR OF ERROR RETURNED
00110 C
00120       COMMON/LIO/LIOA,LIOB,LIOC,LIOD,LIOE,LIOF,LIOG,LIOH,
00130      & LIOI,LIOJ,LIOK,LIOL,LIOM,LION,LIOO,LIOP,LIOQ,LIOR,
00140      & LIOS,LIOT,LIOU,LIOV,LIOW
00150       INTEGER LIOA,LIOB,LIOC,LIOD,LIOE,LIOF,LIOG,LION,
00160      & LIOI,LIOJ,LIOK,LIOL,LIOM,LION,LIOO,LIOP,LIOQ,LIOR,
00170      & LIOS,LIOT,LIOU,LIOV,LIOW
00180 C
00190 C PSA LOGICAL IO UNITS
00200 C
00210 C LIOA - SYNTAX ERRORS FOR UPDATE
00220 C LIOB - PARSE TABLES FOR UPDATE (OUTPUT OF PREPROCESSOR)
00230 C LIOC - DATA BASE
00240 C LIOD - DATA BASE TABLES
00250 C LIOE - INPUT OF NAMES FOR SOME OUTPUTS
00260 C LIOF - PSL INPUT TO UPDATE
00270 C LIOG - OUTPUT OF REPORT SYSTEM
00280 C LIOH - FORMATTED LIST PUNCH OUTPUT
00290 C LIOI - NAMES TO BE SORTED FOR XREF
00300 C LIOJ - SYMBOL TABLE DUMP FOR XREF
00310 C LIOK - COMMAND LANGUAGE INPUT
00320 C LIOL -  SYSTEM ERRORS
00330 C LIOM - COMMAND CORRECTION
00340 C LION - COMMAND LANGUAGE ECHO
00350 C LIOO - COMMENTS TO USER FOR DELETE RENAME ETC.
00360 C LIOP - INPUT/OUTPUT OF STUFF FOR NAME USAGE INDEX
00370 C LIOQ - INPUT/OUTPUT OF COMMON AREAS FOR PARAMETERS
00380 C LIOR - INPUT FOR CHANGE-TYPE COMMAND
00390 C LIOS - INPUT/OUTPUT FOR NAME GEN NAMES
00400 C LIOT - FILE FOR SAVE COMMAND
00410 C LIOU - PUNCH FILE FOR PROCIO COMMAND
00420 C LIOV - INPUT FILE FOR PLONG
00430 C LIOW - PUNCH FILE FOR INDEX
00440 C
00450       INTEGER*2 LEN2
00460       INTEGER LNUM,ATT/2/
00470       INTEGER BUF(20)
00480       INTEGER LEN
00490 C.....................
00500       CALL SMOVE(ERRCH,1,7HPSA000: ,1,7)
00510       LEN = 3
00520       IF(ERRNUM .LT. 100) LEN = 2
00530       IF(ERRNUM .LT. 10) LEN = 1
00540       CALL ITOC(ERRNUM,ERRCH,7-LEN,LEN)
00550       LNUM = ERRNUM * 1000
00560 C     CALL READ(BUF,LEN2,ATT,LNUM,LIOA)
00570       LEN = LEN2
00580 C
00590       DO 10 I=1,ERRNUM
00600       READ(LIOA,20) BUF
00610    10 CONTINUE
00620    20 FORMAT(20A4)
00630       REWIND LIOA
00640 C
00650       DO 30 I=1,79
00660       LEN = 80 - I
00670       J = ISCOMP(BUF,LEN,1H  ,1,1)
00680       IF(J .NE. 0) GOTO 40
00690    30 CONTINUE
00700       LEN = 1
00710 C
00720    40 CALL SMOVE(ERRCH,8,BUF,1,LEN)
00730       ERRLEN = LEN + 7
00740       RETURN
00750       END
READY
```

```
      SUBROUTINE GETTIM(HHMMSS)
C
C RETURN TIME IN CHARACTERS
C
      INTEGER HHMMSS(1)
C
C HHMMSS  -- 8 CHARACTER TIME (HH:MM:SS )
C
C.........................
      CALL CLOCK(HHMMSS)
      RETURN
      END
```

```
list 'ts0409.dbrand.asm(randcl)'
 'TS0409.DBRAND.ASM(RANDCL)'
00010            TITLE 'RANDCL(FDESG) RANDOM CLOSE'
00020 RANDCL     CSECT
00030            USING *,12
00040            STM   14,12,12(13)
00050            LR    12,15
00060            LA    15,CLSA
00070            ST    15,8(,13)
00080            ST    13,4(,15)
00090            LR    13,15
00100 *
00110            L     2,0(,1)    2=DCB ADDR
00120            L     2,0(,2)
00130 *
00140            CLOSE ((2))
00150 *
00160            SR    15,15
00170            L     13,4(,13)
00180            LM    0,12,20(13)
00190            L     14,12(,13)
00200            BR    14
00210 *
00220 CLSA       DS    18A
00230            END
READY
```

```
list 'ts0409.dbrand.asm(randop)'
 'TS0409.DBRAND.ASM(RANDOP)'
00010            TITLE 'RANDOP(LIONUM,FDESC) OPEN A RANDOM FILE FOR UPDATE'
00020 RANDOP     CSECT
00030            USING *,12
00040            STM   14,12,12(13)
00050            LR    12,15
00060            LA    15,OPSA
00070            ST    15,8(,13)
00080            ST    13,4(,15)
00090            LR    13,15
00100 *
00110            LM    2,3,0(1)
00120            L     2,0(,2)
00130            CVD   2,OPDUB
00140            MVC   OPPACK,=X'402120202020'
00150            ED    OPPACK,OPDUB+5
00160            MVC   OPDDNAME+2(4),OPPACK+2
00170            LA    4,OPDCB
00180            USING IHADCB,4
00190            MVC   DCBDDNAM,OPDDNAME
00200            OPEN  (OPDCB,(UPDAT))
00210            LA    1,OPDCB
00220            ST    1,0(,3)      RETURN FDESC = DCB ADDRESS
00230            SR    15,15
00240            L     13,4(,13)
00250            LM    0,12,20(13)
00260            L     14,12(,13)
00270            BR    14
00280 *
00290 OPDUB      DS    D
00300 OPSA       DS    18A
00310 OPPACK     DS    CL6
00320 OPDDNAME   DC    CL8'DB0000'
00330 OPDCB      DCB   BUFNO=1,DSORG=DA,MACRF=(RIC,WIC),OPTCD=R,BLKSIZE=4096,  X
00340                  RECFM=F
00350            LTORG
00360            DCBD  DSORG=DA,DEVD=DA
00370            END
READY
```

```
list 'ts0400.dbrand.asm(randoq)'
 'TS0400.DBRAND.ASM(RANDOQ)'
00010            TITLE 'RANDOQ(LIONUM,FDESC) OPEN A RANDOM FILE FOR CREATE'
00020 RANDOQ     CSECT
00030            USING *,12
00040            STM   14,12,12(13)
00050            LR    12,15
00060            LA    15,OOSA
00070            ST    15,8(,13)
00080            ST    13,4(,15)
00090            LR    13,15
00100 *
00110            L1    2,5,0(1)
00120            L     2,0(,2)
00130            CVD   2,OODUB
00140            UNPK  OOPACK,=X'402120202020'
00150            ED    OOPACK,OODUB+5
00160            MVC   OODDNAME+2(4),OOPACK+2
00170            LA    4,OODCB
00180            USING IHADCB,4
00190            MVC   DCBDDNAM,OODDNAME
00200            OPEN  (OODCB,(OUTPUT))
00210            LA    1,OODCB
00220            ST    1,0(,3)      RETURN FDESC = DCB ADDRESS
00230            SR    15,15
00240            L     13,4(,13)
00250            LM    0,12,20(13)
00260            L     14,12(,13)
00270            BR    14
00280 *
00290 OODUB      DS    D
00300 OOSA       DS    18A
00310 OOPACK     DS    CL6
00320 OODDNAME   DC    CL8'DB0000'
00330 OODCB      DCB   BLKSIZE=4096,BUFNO=1,DSORG=PS,MACRF=(PL),RECFM=F
00340 *                MACRF=(PL),RECFM=F
00350            LTORG
00360            DCBD
00370            END
READY
```

```
list 'ts0409.dbrand.asm(randrd)'
 'TS0409.DBRAND.ASM(RANDRD)'
00010            TITLE 'RANDRD(FDESC,PAGENO,BUFFER) RANDOM READ '
00020 RANDRD     CSECT
00030            USING *,12
00040            STM   14,12,12(13)
00050            LR    12,15
00060            LA    15,RDSA
00070            ST    15,8(,13)
00080            ST    13,4(,15)
00090            LR    13,15
00100 *
00110            LM    2,4,0(1)
00120            L     2,0(,2)    2=DCB ADDR
00130            L     1,0(,3)
00140            BCTR  1,0          CHANGE PAGENO TO REL BLOCK NO
00150            ST    1,RDBLKNO
00160 *          LA    3,RDBLKNO+1
00170 *
00180            READ  RDDECB,DI,(2),(4),'S',0,RDBLKNO+1
00190            CHECK RDDECB
00200 *
00210            SR    15,15
00220            L     13,4(,13)
00230            LM    0,12,20(13)
00240            L     14,12(,13)
00250            BR    14
00260 *
00270 RDSA       DS    18A
00280 RDBLKNO    DS    F
00290            END
READY
```

```
list 'ts0409.dbrand.asm(randrw)'
   'TS0409.DBRAND.ASM(RANDRW)'
  00010            TITLE 'RANDRW(FDESG,PAGENO,BUFFER) RANDOM WRITE'
  00020 RANDRW     CSECT
  00030            USING *,12
  00040            STM   14,12,12(13)
  00050            LR    12,15
  00060            LA    15,RWSA
  00070            ST    15,8(,13)
  00080            ST    13,4(,15)
  00090            LR    13,15
  00100 *
  00110            LM    2,4,0(1)
  00120            L     2,0(,2)    2=DCB ADDR
  00130            L     1,0(,3)
  00140            BCTR  1,0          CHANGE PAGENO TO REL BLOCK NO
  00150            ST    1,RWBLKNO
  00160            LA    3,RWBLKNO+1
  00170 *
  00180            WRITE RWDECB,DI,(2),(4),'S',0,(3)
  00190            CHECK RWDECB
  00200 *
  00210            SR    15,15
  00220            L     13,4(,13)
  00230            LM    0,12,20(13)
  00240            L     14,12(,13)
  00250            BR    14
  00260 *
  00270 RWSA       DS    18A
  00280 RWBLKNO    DS    F
  00290            END
  READY
```

```
list 'ts0409.dbrand.asm(randwt)'
 'TS0409.DBRAND.ASM(RANDWT)'
00010           TITLE 'RANDWT(FDESG,PAGENO,BUFFER) RANDOM WRITE'
00020 RANDWT    CSECT
00030           USING *,12
00040           STM   14,12,12(13)
00050           LR    12,15
00060           LA    15,WTSA
00070           ST    15,8(,13)
00080           ST    13,4(,15)
00090           LR    13,15
00100 *
00110           LM    2,4,0(1)
00120           L     2,0(,2)    2=DCB ADDR
00130 *
00140           WRITE WTDECB,SF,(2),(4),'S'
00150           CHECK WTDECB
00160 *
00170           SR    15,15
00180           L     13,4(,13)
00190           LM    0,12,20(13)
00200           L     14,12(,13)
00210           BR    14
00220 *
00230 WTSA      DS    18A
00240           END
READY
```

APPENDIX II

LISTING OF TIMER SUBROUTINES

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                        F01MAY72  6/23/75

                                    2 ************************************************************************
                                    3 *   CALL INTIME(KEY)                                                   *
                                    4 *        KEY     =0  RESULT IN 26.04166 MICRO SECOND UNITS             *
                                    5 *        KEY     =1  RESULT IN .01 SECOND UNITS                        *
                                    6 *                                                                       *
                                    7 *   INTVL = JTIME(ITOTAL,KLOCK)                                        *
                                    8 *        INTVL   CPU TIME USED SINCE LAST USE OF TIMING ROUTINE         *
                                    9 *                  IN UNITS SPECIFIED BY KEY                            *
                                   10 *        ITOTAL  TOTAL TIME USED SINCE CALLING INTIME                   *
                                   11 *                  IN UNITS SPECIFIED BY KEY                            *
                                   12 *        KLOCK   CLOCK TIME REMAINING (26 MICRO SEC UNITS)              *
                                   13 ************************************************************************
                                   14          ENTRY INTIME,JTIME
000000                             15          USING *,12
000000 47FF 000C          0000C    16 INTIME   BC    15,12(15)
000004 06                          17          DC    AL1(6)
000005 C9D5E3C9D4C5                18          DC    CL6'INTIME'
00000B 00
00000C 90EC D00C          0000C    19          STM   14,12,12(13)    SAVE REGISTERS
000010 18CF                        20          LR    12,15           ESTABLISH BASE REGISTER
000012 1833                        21          SR    3,3
000014 5030 C080          00080    22          ST    3,TOT           SET TOTAL ELAPSED TIME TO 0
000018 5830 C0A8          000A8    23          L     3,V(3)          LOAD 3 FOR SETTING CLOCK
00001C 5030 C084          00084    24          ST    3,CLOCK         INITIALIZE THE CLOCK
000020 5030 C0AC          000AC    25          ST    3,LASTIM
000024 5811 0000          00000    26          L     1,0(1)          LOAD ADDRESS OF ARGUMENT
000028 5811 0000          00000    27          L     1,0(1)          LOAD KEY
00002C 1211                        28          LTR   1,1             IS KEY 0 OR 1
00002E 4780 C03A          0003A    29          BC    8,STIME         BRANCH IF KEY=0
000032 4150 0036          00036    30          LA    5,R-JTIME       SET BRANCH LOCATION IN JTIME ROUTINE
000036 4250 C081          00081    31          STC   5,BRANCH+3      MODIFY BRANCHING INSTRUCTION
                                   32 STIME    STIMER TASK,TUINTVL=CLOCK
00003A 4110 C084          00084    33*STIME    LA    1,CLOCK LOAD PARAMETER REG 1
00003E 1800                        34*         SR    0,0 INDICATE TASK,TUINTVL=
000040 0A2F                        35*         SVC   47 ISSUE STIMER SVC
000042 98EC D00C          0000C    36          LM    14,12,12(13)    RESET REGISTERS
000046 92FF D00C    0000C         37          MVI   12(13),X'FF'
00004A 07FE                        38          BCR   15,14           RETURN TO CALLING PROGRAM

00004C                             40          USING *,12
00004C 47FF 000A          0000A    41 JTIME    BC    15,10(15)
000050 05                          42          DC    AL1(5)
000051 D1E3C9D4C5                  43          DC    CL5'JTIME'
000056 90EC D00C          0000C    44          STM   14,12,12(13)
00005A 18CF                        45          LR    12,15           ESTABLISH BASE REGISTER
00005C 9867 1000          00000    46          LM    6,7,0(1)        ADDRESSES OF ARGUMENTS
                                   47          TTIMER
000060 1B11                        48*         SR    1,1 INDICATE TIME REMAINING - NO CANCEL      20272
000062 0A2E                        49*         SVC   46 ISSUE TTIMER                              20272
000064 5830 C060          000AC    50          L     3,LASTIM        LOAD LAST CLOCK TIME
000068 5007 0000          00000    51          ST    0,0(7)          RETURN CURRENT VALUE OF CLOCK
00006C 5000 C060          000AC    52          ST    0,LASTIM        SAVE CURRENT CLOCK TIME
000070 1850                        53          LR    5,0             GPR5=CURRENT CLOCK TIME LEFT
```

```
000072 1835                          54          SR    3,5             PREVIOUS MINUS CURRENT CLOCK TIME
000074 1853                          55          LR    5,3             TIME INTERVAL IN CLOCK UNITS
000076 5A30 C064             00080   56          A     3,TOT           COMPUTE TOTAL ELAPSED TIME
00007A 5030 C064             00080   57          ST    3,TOT           STORE TOTAL TIME
00007E 47F0 C042             00046   58 BRANCH   BC    15,C            BRANCH TO B OR C DEPENDING ON KEY
000082 1B22                          59 B        SR    2,2
000084 5D20 C058             000A4   60          D     2,N384          CONVERT TOTAL TIME TO .01 SEC UNITS
000088 1B44                          61          SR    4,4
00008A 5D40 C058             000A4   62          D     4,N384          CONVERT INTERVAL TO .01 SEC UNITS
00008E 1805                          63 C        LR    0,5             RETURN DURATION OF LAST INTERVAL
000090 5036 0000             00000   64          ST    3,0(6)          STORE TOTAL IN CALLING PROGRAM
000094 981C D018             00018   65          LM    1,12,24(13)     RESTORE REGISTERS
000098 58ED 000C             0000C   66          L     14,12(13)
00009C 92FF D00C       0000(13)      67          MVI   12(13),X'FF'
0000A0 07FE                          68          BCR   15,14           RETURN
0000A2 0000
0000A4 00000180                      69 N384     DC    F'384'          CONVERSION FACTOR FOR .01 SEC
0000A8 7FFFFFFF                      70 N231     DC    X'7FFFFFFF'     2**31 - 1
0000AC                               71 LASTIM   DS    1F
0000B0                               72 TOT      DS    1F
000084                               73 CLOCK    DS    1F
                                     74          END
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                        F01MAY72  6/23/75

                                    1 *
                                    2 * FORTRAN TIME SUBROUTINE   --- COURTESY R. T. JONES
                                    3 * ALIASES: CLOCK,DATE,ITIME,ITIMER; NAME: FTIME(R)
                                    4 *
000000                              5 FTIME   CSECT
000000 47F0 F00C            0000C   6         B     12(0,15)
000004 08                           7         DC    X'08'
000005 C6E3C904C54040               8         DC    C'FTIME  '
00000C 07FE                         9         BR    14
                                   10 *
                                   11 *                  SUBROUTINE CLOCK(A64BIT)                          0159
                                   12 *                        REFERENCE - ED MAYBURY'S TIMER (JOBTAG18)   0160
                                   13 *  THIS SUBROUTINE WILL RETURN THE TIME AS EIGHT EBCDIC CHARACTERS IN  0161
                                   14 *  THE ARGUMENT IN THE FORM HH.MM.SS WHERE HH=HRS, MM=MINS, AND SS=SECS 0162
                                   15 *                                                                      0163
                                   16        ENTRY CLOCK                                                     0164
00000E 0700                        17        CNOP  0,4                                                       0165
000010                             18 CLOCK  EQU   *                                                         0166
000010 90EC 000C            0000C  19        STM   14,12,12(13)    SAVE ALL REGS EXCEPT R13 IN CALLING PSA   0167
000014 05C0                        20        BALR  12,0            USE BASE REGISTER 12 FOR PROGRAM PROPER   0168
000016                             21        USING *,12                                                      0169
000016 1890                        22          LR    9,13
000018 4100 C20E            00224  23          LA    13,SAVAREA
00001C 5009 0008            00008  24          ST    13,8(9)
000020 5090 C212            00228  25          ST    9,SAVAREA+4
                                   26 *                                                                      0171
000024 5840 1000            00000  27        L     4,0(0,1)        FETCH ADDRESS OF ARGUMENT ONE             0172
                                   28        TIME  DEC             GET TIME IN R0,GET DATE IN R1             0173
000028 4110 0002            00002  29+         LA    1,2(0,0) LOAD 1 TO SPECIFY UNIT
00002C 0A08                        30+         SVC   11 ISSUE TIME SVC
00002E 5000 C04A            00060  31        ST    0,PACKED              SAVE PACKED TIME                    0174
000032 F363 C052 C04A 00068 00060  32        UNPK  UNPAKD(7),PACKED(4)   UNPACK TIME TO ZHZHZMZMZSZSHT      0175
000038 D201 C05A C052 00070 00068  33        MVC   A64BIT(2),UNPAKD      GET HH                              0176
00003E D201 C050 C054 00073 0006A  34        MVC   A64BIT+3(2),UNPAKD+2  GET MM                              0177
000044 D201 C060 C056 00076 0006C  35        MVC   A64BIT+6(2),UNPAKD+4  GET SS                              0178
00004A D207 4000 C05A 00000 00070  36        MVC   0(8,4),A64BIT         STORE ARGUMENT IN CALLING PROGRAM   0179
                                   37 *
                                   38 * RETURN
                                   39 *
000050 58D0 C212            00228  40          L     13,SAVAREA+4
000054 58EE 000C            0000C  41          L     14,12(13)
000058 982C 001C            0001C  42          LM    2,12,28(13)
00005C 07FE                        43          BR    14              RETURN TO CALLING PROGRAM


                                   45 *                  DATA RESERVATIONS AND CONSTANTS                   0182
00005E 0700                        46        CNOP  0,4                                                       0183
000060                             47 PACKED DS    1F                PACKED TIME                           0184
000068                             48 UNPAKD DS    1D                UNPACKED TIME                         0185
000070 F0F048F0F04BF0F0            49 A64BIT DC    C'00.00.00'       OUTPUT ARGUMENT (64 BITS)             0186
```

```
 LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                  F01MAY72  6/23/75

                                    51 *                    SUBROUTINE DATE (B64BIT)                              0188
                                    52 *  THIS SUBROUTINE WILL RETURN THE DATE AS EIGHT EBCDIC CHARACTERS IN      0189
                                    53 *  THE ARGUMENT IN THE FORM MM/DD/YY WHERE MM=MONTH, DD=DAY, YY=YEAR.      0190
                                    54 *                          REFERENCE - ED MAYBURY'S GETSD (JD8TAG13)     0191
                                    55 *                THE FOLLOWING SECTION IS TAKEN FROM                     0192
                                    56 *                ED MAYBURY'S COBOL COMPATIBLE                           0193
                                    57 *                ASM ROUTINE 'GETSD'.                                    0194
                                    58 *  FIXED TO HANDLE SUBSEQUENT CALLS 2-13-73 JPH D72
                                    59 *                                                                        0195
                                    60         ENTRY DATE                                                       0196
000078                              61         CNOP  0,4                                                        0197
000078                              62 DATE    EQU   *                                                          0198
000078 90EC D00C          0000C     63         STM   14,12,12(13)    SAVE ALL REGS EXCEPT R13 IN CALLING PSA    0199
00007C 05C0                         64         BALR  12,0            USE BASE REGISTER 12 FOR PROGRAM PROPER    0200
00007E                              65         USING *,12                                                       0201
00007E 189D                         66           LR    9,13
000080 41D0 C1A6          00224     67           LA    13,SAVAREA
000084 50D9 0008          00008     68           ST    13,8(9)
000088 5090 C1AA          00228     69           ST    9,SAVAREA+4
                                    70 *                                                                        0203
00008C 5840 1000          00000     71         L     4,0(0,1)        FETCH ADDRESS OF ARGUMENT ONE              0204
                                    72         TIME  DEC                                                        0205
000090 4110 0002          00002     73+          LA    1,2(0,0) LOAD 1 TO SPECIFY UNIT
000094 0A0B                         74+          SVC   11 ISSUE TIME SVC
000096 5010 C12A          001A8     75         ST    1,DATAREA       SAVE PACKED DATE                           0206
00009A 1822                         76         SR    2,2     CLEAR REG 2 FOR 2-3 DIVIDE COMING UP               0207
00009C D100 C131 C12B 00IAF 00IA9   77           MVN   FW+3(1),DATAREA+1     MOVE LOWER 4 BITS OF YEAR        0208
                                    78 *                                        I.E. , ONES DIGIT          01 -0208
0000A2 5830 C12E          00IAC     79         L     3,FW                                                       0209
0000A6 9110 C12B          00IA9     80           TM    DATAREA+1,X'10'  IS TENS DIGIT ODD?
0000AA 4780 C034          00082     81           BZ    DIVIDE   NO,GO TO DIVIDE BY 4
0000AE 4133 0002          00002     82           LA    3,2(3)   YES ADD 2 TO YEAR DIGIT TO COMPENSATE FOR ODD
000082                              83 DIVIDE  EQU   *        TENS DIGIT, THEN GO TO DIVIDE BY 4
000082 D705 C13A C13A 00188 00188   84           XC    FW1(6),FW1       ZERO OUT FW1 FOR SUBSEQUENT CALL JPH
000088 D203 C11E C1F2 00I9C 00270   85           MVC   TABC(4),=F'1'    RESTORE F'1' TO TABC FOR SUBSEQUENT CALL
00008E D201 C142 C1F6 001C0 00274   86           MVC   SAVI(2),=H'0'    ZERO OUT SAVI FOR SUBSEQUENT CALL JPH
0000C4 5D20 C132          00180     87         D     2,CONST          DIVIDE YR BY 4                             0210
0000C8 1222                         88         LTR   2,2          TEST FOR REMAINDER                           0211
0000CA 4780 C058          000D6     89         BZ    LPYR         LEAP YEAR                                    0212
0000CE 4820 C136          00184     90         LH    2,CONST1                                                  0213
0000D2 4020 C0F6          00174     91         STH   2,MDTAB+2    ADJUST FOR NON LEAP YEAR                     0214
0000D6 D201 C140 C12C 0018E 00IAA   92 LPYR    MVC   FW1+6(2),DATAREA+2    MOVE IN DDD                      0215
0000DC 4F20 C13A          00188     93         CVB   2,FW1        CONVERT DDD TO BINARY                        0216
0000E0 4130 C0F4          00172     94         LA    3,MDTAB      TABLE POINTER                                0217
0000E4 4820 3000          00000     95 LOOP    SH    2,0(0,3)     SUBTRACT ITEM FROM DDD                       0218
0000E8 D201 C142 3000 001C0 00000   96         MVC   SAVI(2),0(3)                                              0219
0000EE 1222                         97         LTR   2,2                                                       0220
0000F0 4780 C0C0          0013E     98         BZ    FINI                                                      0221
0000F4 4720 C0C8          00146     99         BP    PLUS                                                      0222
0000F8 4A20 C142          001C0    100         AH    2,SAVI       MINUS                                        0223
0000FC 4E20 C13A          00188    101 CVDI    CVD   2,FW1        CONVERT TO PACKED DECIMAL                    0224
000100 F321 C0EC C140 00I6A 0018E   102         UNPK  LOCN+2(3),FW1+6(2)                                        0225
000106 96F0 C0EE          0016C     103         OI    LOCN+4,X'F0'                                              0226
00010A 5820 C11E          00I9C     104         L     2,TABC                                                    0227
00010E 4E20 C11A          00198     105         CVD   2,TABC1                                                   0228
```

```
 LOC   OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                  F01MAY72  6/23/75

000112 F321 C0E9 C120 00167 0019E   106          UNPK  LOCN-1(3),TABC+2(2)
000118 96F0 C0EB      00169         107          OI    LOCN+1,X'F0'                                    0230
00011C F321 C0F0 C128 0016E 001A9   108          UNPK  LOCN+6(3),DATAREA+1(2)  AND YR               0231
000122 D207 4000 C0EA 00000 00168   109          MVC   0(8,4),LOCN      MOVE INTO USERS AREA       0232
000128 9261 4002      00002         110          MVI   2(4),X'61'       /                          0233
00012C 9261 4005      00005         111          MVI   5(4),X'61'       /                          0234
                                    112 *
                                    113 * RETURN
                                    114 *
000130 58D0 C1AA            00228   115            L     13,SAVAREA+4
000134 58ED 000C            0000C   116            L     14,12(13)
000138 982C D01C            0001C   117            LM    2,12,28(13)
00013C 07FE                         118            BR    14              RETURN TO CALLING PROGRAM
```

APPENDIX III

DATA SET CHARACTERISTICS AND STORAGE REQUIREMENTS

This Appendix lists the URL/URA data sets with their respective file organization and record formats (in OS/TSO notation), logical and physical record length (in bytes) and track requirements as they were when copied from disk to the distribution tape. These data set attributes may be required for installations which do not handle IBM standards labelled tapes.

For object distribution:

DATA SET NAME	DATA SET ORGANIZATION	RECORD FORMAT	RECORD LENGTH	BLOCK SIZE	NO. OF TRACKS (2314)
TSØ448.CALLIB.LOAD	PO	U		1024	759
TSØ448.PSADDL.DATA	PS	FB	80	8000	1
TSØ448.SECRET.DATA	PS	FB	80	8000	2
TSØ448.HELPSHRT.DATA	PS	FB	80	8000	3
TSØ448.HELPLONG.DATA	PS	FB	80	8000	11
TSØ448.PSAERROR.DATA	PA	FB	80	800	3
TSØ448.URA.CLIST	PS	FB	80	80	2
TSØ448.CLI.CLIST	PS	F	80	80	1
TSØ448.DDLA.CLIST	PS	VB	255	1680	1
TSØ448.DBIN.CLIST	PS	VB	255	1680	1
TSØ448.PSALINK.CLIST	PS	VB	255	1680	1
TSØ448.PSALINK1.CLIST	PS	VB	255	1680	1
TSØ448.NCLI.CLIST	PS	VB	255	1680	1
TSØ448.LIBMTS.FORT	PO	FB	80	6400	2
TSØ448.DBRAND.ASM	PO	FB	80	6400	2
TSØ448.DBLIB.LOAD	PO	U		13030	21
TSØ448.FLIB.LOAD	PO	U	80	13030	32
TSØ448.SLIB.LOAD	PO	U		1024	5
TSØ448.CLI.OBJ	PS	FBS	80	400	19
TSØ448.CLIEX.OBJ	PS	FBS	80	400	6
TSØ448.CM.OBJ	PS	FBS	80	400	2
TSØ448.CNC.OBJ	PS	FBS	80	400	2
TSØ448.CT.OBJ	PS	FBS	80	400	2
TSØ448.CONT.OBJ	PS	FBS	80	400	2
TSØ448.DBS.OBJ	PS	FBS	80	400	2
TSØ448.DCOM.OBJ	PS	FBS	80	400	2
TSØ448.DEL.OBJ	PS	FBS	80	400	2
TSØ448.DICT.OBJ	PS	FBS	80	400	2
TSØ448.DP.OBJ	PS	FBS	80	400	4
TSØ448.DPSL.OBJ	PS	FB	80	400	12
TSØ448.EI.OBJ	PS	FBS	80	400	2

DATA SET NAME	DATA SET ORGANIZATION	RECORD FORMAT	RECORD LENGTH	BLOCK SIZE	NO. OF TRACKS
TSØ448.FPS.OBJ	PS	FBS	80	400	16
TSØ448.FREQ.OBJ	PS	FBS	80	400	1
TSØ448.IDX.OBJ	PS	FBS	80	400	1
TSØ448.KPER.OBJ	PS	FBS	80	400	1
TSØ448.KPRT.OBJ	PS	FBS	80	400	1
TSØ448.NGA.OBJ	PS	FB	80	400	4
TSØ448.NGP.OBJ	PS	FB	80	400	1
TSØ448.NLA.OBJ	PS	FB	80	400	1
TSØ448.NLP.OBJ	PS	FB	80	400	1
TSØ448.PAU.OBJ	PS	FB	80	400	1
TSØ448.PCOM.OBJ	PS	FBS	80	400	2
TSØ448.PIC.OBJ	PS	FBS	80	400	9
TSØ448.PRIO.OBJ	PS	FB	80	400	3
TSØ448.RCOM.OBJ	PS	FB	80	400	2
TSØ448.REN.OBJ	PS	FB	80	400	2
TSØ448.SUM.OBJ	PS	FBS	80	400	1
TSØ448.SYNU.OBJ	PS	FB	80	400	16
TSØ448.XREF.OBJ	PS	FB	80	400	1
TSØ448.DBIN.OBJ	PS	FBS	80	400	1
TSØ448.DDLA.OBJ	PS	FBS	80	400	4
TSØ448.PSL.OBJ	PS	FB	80	400	9
TSØ448.PSLBLK.OBJ	PS	FB	80	400	1
TSØ448.LIBBLK.OBJ	PS	FB	80	400	1
TSØ448.DBBLK.OBJ	PS	FB	80	400	1
TSØ448.EXA.DATA	PS	FB	80	800	1
TSØ448.EXB.DATA	PS	FB	80	800	1

For source distribution:

DATA SET NAME	DATA SET ORGANIZATION	RECORD FORMAT	RECORD LENGTH	BLOCK SIZE	NO. OF TRACKS
TSØ448.PSADDL.DATA	PS	FB	80	8000	1
TSØ448.SECRET.DATA	PS	FB	80	8000	2
TSØ448.HELPSHRT.DATA	PS	FB	80	8000	3
TSØ448.HELPLONG.DATA	PS	FB	80	8000	11
TSØ448.PSAERROR.DATA	PS	FB	80	800	3
TSØ448.DDLA.DATA	PS	FB	80	8000	6
TSØ448.DBIN.DATA	PS	FB	80	8000	1
TSØ448.INCL.DATA	PS	FB	80	8000	2
TSØ448.DBCOM.DATA	PS	FB	80	8000	1
TSØ448.PSACOM.DATA	PS	FB	80	8000	19

DATA SET NAME	DATA SET ORGANIZATION	RECORD FORMAT	RECORD LENGTH	BLOCK SIZE	NO. OF TRACKS
TSØ448.EXA.DATA	PS	FB	80	800	1
TSØ448.EXB.DATA	PS	FB	80	800	1
TSØ448.SLIB.ASM	PO	FB	80	6400	4
TSØ448.DBASM.ASM	PO	FB	80	6400	1
TSØ448.DBRAND.ASM	PO	FB	80	6400	2
TSØ448.LIBMTS.FORT	PO	FB	80	6400	2
TSØ448.DBUSER.DATA	PS	FB	80	8000	41
TSØ448.DBLOW.DATA	PS	FB	80	8000	15
TSØ448.DBTAB.DATA	PS	FB	80	8000	4
TSØ448.FLIB.DATA	PS	FB	80	8000	34
TSØ448.PSL.DATA	PS	FB	80	8000	20
TSØ448.PSLBLK.DATA	PS	FB	80	8000	3
TSØ448.DBBLK.DATA	PS	FB	80	8000	1
TSØ448.LIBBLK.DATA	PS	FB	80	8000	2
TSØ448.CLI.DATA	PS	FB	80	8000	42
TSØ448.CLIEX.DATA	PS	FB	80	8000	13
TSØ448.CM.DATA	PS	FB	80	8000	4
TSØ448.CNC.DATA	PS	FB	80	8000	4
TSØ448.CONT.DATA	PS	FB	80	8000	2
TSØ448.CT.DATA	PS	FB	80	8000	2
TSØ448.DBS.DATA	PS	FB	80	8000	2
TSØ448.DCOM.DATA	PS	FB	80	8000	1
TSØ448.DEC.DATA	PS	FB	80	8000	2
TSØ448.DICT.DATA	PS	FB	80	8000	2
TSØ448.DP.DATA	PS	FB	80	8000	8
TSØ448.DPSL.DATA	PS	FB	80	8000	30
TSØ448.EI.DATA	PS	FB	80	8000	4
TSØ448.FPS.DATA	PS	FB	80	8000	21
TSØ448.FREQ.DATA	PS	FB	80	8000	1
TSØ448.IDX.DATA	PS	FB	80	8000	1
TSØ448.KPER.DATA	PS	FB	80	8000	1
TSØ448.KPRT.DATA	PS	FB	80	8000	1
TSØ448.NGA.DATA	PS	FB	80	8000	6
TSØ448.NGP.DATA	PS	FB	80	8000	1
TSØ448.NLA.DATA	PS	FB	80	8000	1
TSØ448.NLP.DATA	PS	FB	80	8000	1
TSØ448.PAV.DATA	PS	FB	80	8000	1
TSØ448.PCOM.DATA	PS	FB	80	8000	2
TSØ448.PIC.DATA	PS	FB	80	8000	15
TSØ448.PRIO.DATA	PS	FB	80	8000	3
TSØ448.RCOM.DATA	PS	FB	80	8000	3
TSØ448.REN.DATA	PS	FB	80	8000	1
TSØ448.STR.DATA	PS	FB	80	8000	3
TSØ448.SUM.DATA	PS	FB	80	8000	1

DATA SET NAME	DATA SET ORGANIZATION	RECORD FORMAT	RECORD LENGTH	BLOCK SIZE	NO. OF TRACKS
TSØ448.SYNU.DATA	PS	FB	80	8000	34
TSØ448.XREF.DATA	PS	FB	80	8000	1
TSØ448.PSACOM.DATABASE	PS	F		4096	17
TSØ448.FORTIHC.CLIST	PS	VB	255	1680	1
TSØ448.LIBBLK.OBJ	PS	FB	80	400	1
TSØ448.DBBLK.OBJ	PS	FB	80	400	1
TSØ448.PSLBLK.OBJ	PS	FB	80	400	1
TSØ448.PSL.OBJ	PS	FB	80	400	9
TSØ448.DBLIB.LOAD	PO	U		13030	21
TSØ448.PUB.LOAD	PO	U	80	13030	32
TSØ448.SUB.LOAD	PO	U		1024	5

REFERENCES

1. Ernest Allen Hershey III and Michel J. Bastarache, "The Structure and Contents of a PSA Data Base," ISDOS Working Paper No. 87, University of Michigan, November 1974.

2. Ernest A. Hershey III, "A Data Base Management System for PSA Based on DBTG 71," ISDOS Working Paper No. 88, University of Michigan, September 1973.

3. Ernest Allen Hershey III, Daniel Teichroew, Douglas L. R. Berg, Edward Winters, Michel J. Bastarache, R. L. Davies, and Claudia R. Stallings, "User Requirements Language Reference Manual," ISDOS Research Project, University of Michigan, July 1974.

4. Douglas L. R. Berg, Ernest A. Hershey III, and Michel J. Bastarache, "Problem Statement Analyzer Command Descriptions," ISDOS Working Paper No. 91, University of Michigan, April 1974.